Tutorial ANM: Week 5

Friday, October 6th, 2016

Finite Difference Solver of a Poisson Equation in Two Dimensions

The objective of this assignment is to guide the student to the development of a finite difference method (FDM) solver of a Poisson Equation in two dimensions from scratch. This assignment extends the previous assignment from one to two dimensions. It therefore allows to include scenarios that are more relevant to the practice of a chemical engineering with a passion for numerical simulations. The assignment again consists of both pen-and-paper and implementation exercises.

1 Problem Formulation

The discretized Poisson Equation on the unit square with Dirichlet boundary conditions will be used as an example. Let the unit square be denoted by Ω, i.e., $\Omega = (0,1) \times (0,1)$. Let the boundary $\partial\Omega$ of Ω be denoted by Γ. Let $(x,y) \in \Omega$ denote the dependent variables. Let us assume that the source function $f(x,y)$ with domain Ω is given. We set out to find the function $u(x,y)$ that is solution of the partial differential equation (PDE)

$$-\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} = f(x,y) \text{ for } (x,y) \in \Omega \tag{1}$$

as well as the homogeneous Dirichlet boundary conditions

$$u = 0 \text{ on } \Gamma = \partial\Omega. \tag{2}$$

The extension to more general PDEs and/or boundary conditions is left as an exercise. The *Laplacian* of u is the function

$$\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = \nabla \cdot \nabla u = \operatorname{div} \operatorname{grad} u\,. \tag{3}$$

With this notation the PDE can be written as

$$-\Delta u = f(x,y) \text{ for } (x,y) \in \Omega\,. \tag{4}$$

This equation is called the Poisson equation. In case that $f(x,y) = 0$ it is called the Laplace equation. For particular choices of the source function $f(x,y)$ this PDE supplied with boundary conditions can be solved analytically using separation of variables for instances. In this assignment we will choose $f(x,y)$ such that a given function $u(x,y)$ is the solution of the problem.

Pen and Paper Assignment 1 Choose the function $f(x,y)$ such that the function $u_{ex}(x,y) = x\,(x-1)\,y\,(1-y)$ is the exact solution of the above problem. Do so by computing $-\Delta u_{ex}$. Use the `meshgrid` command in Matlab to plot the function $u(x,y)$ for $0 \leq x \leq 1$ and $0 \leq y \leq 1$ for future reference. Other possible choices for u_{ex} include functions of the form $u_{ex}(x,y) = x\,(x-1)\,y\,(1-y)\,\widehat{u}_{ex}(x,y)$ where $\widehat{u}_{ex}(x,y)$ is a twice differentiable (i.e. sufficiently smooth) function of x and y.

2 Some Mathematics

The Kronecker product will be valuable in constructing the matrix A.

Kronecker Product of Matrices The Kronecker product of two matrices is defined as folllows. Given two rectangular matrices $R_1 \in \mathbb{R}^{m_1 \times n_1}$ and $R_2 \in \mathbb{R}^{m_2 \times n_2}$ its Kronecker product denoted by $R_1 \otimes R_2 \in \mathbb{R}^{m_1 m_2 \times n_1 n_2}$ is formed by replacing each entry $r_{1,ij}$ of R_1 by $r_{1,ij} \cdot R_2$.

We for instance have that

$$\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix} \otimes \begin{bmatrix}0 & 5\\6 & 7\end{bmatrix} = \begin{bmatrix}1\cdot 0 & 1\cdot 5 & 2\cdot 0 & 2\cdot 5\\1\cdot 6 & 1\cdot 7 & 2\cdot 6 & 2\cdot 7\\3\cdot 0 & 3\cdot 5 & 4\cdot 0 & 4\cdot 5\\3\cdot 6 & 3\cdot 7 & 4\cdot 6 & 4\cdot 7\end{bmatrix} = \begin{bmatrix}0 & 5 & 0 & 10\\6 & 7 & 12 & 14\\0 & 15 & 0 & 20\\18 & 21 & 24 & 28\end{bmatrix}$$

The Kronecker product is implemented in the Matlab `kron` command.

Pen and Paper Assignment 2 Given

$$I = \begin{pmatrix}1 & 0\\0 & 1\end{pmatrix} \text{ and } A = \begin{pmatrix}2 & -1\\-1 & 2\end{pmatrix} \tag{5}$$

compute both $A \otimes I$ and $I \otimes A$.

3 Finite Difference Discretization

Discretization of the Geometry For the discretization of the two-dimensional model problem we introduce a mesh size $h = \frac{1}{N}$ (N being the number of elements in one direction) and an uniform grid G_h consisting of $(N+1)^2$ nodes if we include those on the boundary Γ

$$G_h = \{(x_i, y_j) | x_i = (i-1)\,h, y_j = (j-1)\,h; h = \frac{1}{N}, \; 1 \leq i,j \leq N+1; N \in \mathbb{N}\}\,. \tag{6}$$

In this numbering the indices $i = 1$ and $i = N+1$ ($j = 1$ and $j = N+1$) correspond to grid points on the left and right (bottom and top) boundary, respectively. In Figure 1 we illustrate this numbering in the case the $N = 5$.

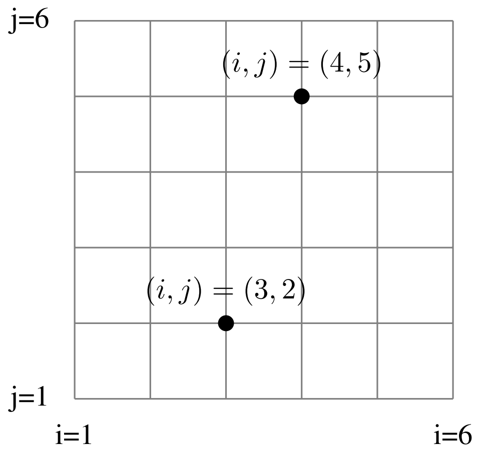

Figure 1: grid ordering using (i,j)

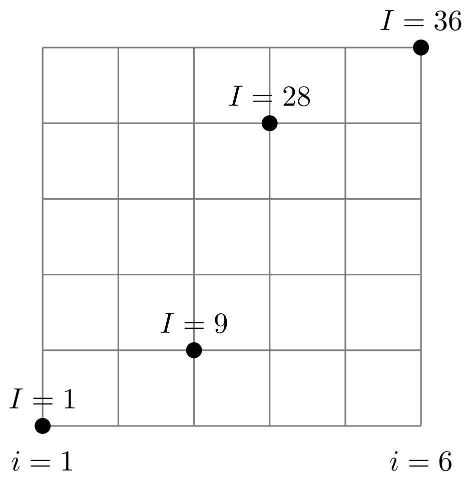

Figure 2: x-lexicografic using I

Discretization of the Physics On G_h we introduce grid vectors approximating the source function $f(x,y)$ and unknown the $u(x,y)$ in the grid nodes with increasing accuracy as $h \to 0$, i.e.,

$$f^h_{i,j} \approx f(x_i, y_j) \text{ for } (x_i, y_j) \in G_h\,, \tag{7}$$

and similarly for $u^h_{i,j}$. The discrete set of data corresponds to what the `meshgrid` command in Matlab generates.

Internal Nodes We enforce the model problem to hold in each grid node and approximate the continuous second order derivatives by central finite difference approximations. The use of finite difference here is generic, as low-order finite element or finite volume discretization result in the same linear system. Using nearest neighbours we have that for the internal nodes that

$$\frac{\partial^2 u}{\partial x^2}(x_i, y_j) = \frac{u^h_{i-1,j} - 2u^h_{i,j} + u^h_{i+1,j}}{h^2} + \mathcal{O}(h^2) \text{ for } 2 \le i,j \le N \tag{8}$$

(and similar for the y-derivative). The approximation to the partial differential equation (1) discretized on internal points of G_h can be written as

$$\frac{-u^h_{i,j-1} - u^h_{i-1,j} + 4u^h_{i,j} - u^h_{i+1,j} - u^h_{i,j+1}}{h^2} = f^h_{i,j} \text{ for } 2 \le i,j \le N\,. \tag{9}$$

The discrete Laplacian on these nodes can then be represented by a so-called *stencil* notation

$$\frac{1}{h^2}\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \tag{10}$$

in which the middle row (column) represents the coupling of the unknown with its left and right (top and bottom) neighbours. This stencil is referred to as the *5-point* stencil.

Boundary Nodes to enforce that the discrete problem satisfies the Dirichlet boundary conditions. One can either add an equation for each node on the Dirichtlet boundary by imposing for these nodes the stencil

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{11}$$

and overwriting $f^h_{i,j}$ on the boundary by 0. Together with the equations on the internal nodes, this results in $(N+1)^2$ linear equations for the $(N+1)^2$ unknowns $\{u^h_{i,j} | 1 \le i,j \le N+1\}$.

4 Linear System Formulation

To arrive at a linear system for the grid unknowns, we introduce a *global* ordering of the grid nodes. We introduce an x-lexicografic ordering of the internal and boundary nodes in which

$$\text{node } (i,j) \text{ is assigned global index } I = i + (j-1)(N+1) \text{ for } 1 \le i,j \le N+1\,, \tag{12}$$

such that $1 \le I \le (N+1)^2$ as shown in Figure 2. This allows us to group the known and unknown grid values $f^h_{i,j}$ and $u^h_{i,j}$ into vectors $\mathbf{u}^h$ and $\mathbf{u}^h$ of size $(N+1)^2$

$$\mathbf{u}^h = \begin{pmatrix} u^h_{1,1} \\ u^h_{1,2} \\ \vdots \\ u^h_{1,N+1} \\ u^h_{2,1} \\ u^h_{2,2} \\ \vdots \\ u^h_{2,N+1} \\ \vdots \\ u^h_{N+1,1} \\ u^h_{N+1,2} \\ \vdots \\ u^h_{N+1,N+1} \end{pmatrix} \in R^{(N+1)^2} \text{ and } \mathbf{f}^h = \begin{pmatrix} f^h_{1,1} \\ f^h_{1,2} \\ \vdots \\ f^h_{1,N+1} \\ f^h_{2,1} \\ f^h_{2,2} \\ \vdots \\ f^h_{2,N+1} \\ \vdots \\ f^h_{N+1,1} \\ f^h_{N+1,2} \\ \vdots \\ f^h_{N+1,N+1} \end{pmatrix} \in R^{(N+1)^2}\,. \tag{13}$$

Changing a matrix in the (i,j) grid numbering to a vector in the I x-lexicografic orderning can be accomplished in Matlab using the `reshape` command.

The model problem then translates into a linear system of equations

$$A^h \, \mathbf{u}^h = \mathbf{f}^h \,, \tag{14}$$

in which the system matrix A^h, $\mathbf{f}^h$ and $\mathbf{u}^h$ represents the discretized differential operator, the discrete source term and the discrete source vector, respectively.

To specify A^h, we introduce four auxiliary matrices. We denote by I^h_{N+1} the identity matrix on $R^{(N+1)\times(N+1)}$, by $\widehat{I}^h$ and T^h the matrices

$$\widehat{I}^h = \begin{pmatrix} 0 & 0 & 0 \\ 0 & I^h_{N-1} & 0 \\ 0 & 0 & 0 \end{pmatrix} \in R^{(N+1)\times(N+1)} \text{ and } T^h = \begin{pmatrix} 4 & -1 & 0 & \dots & \dots & 0 \\ -1 & 4 & -1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & 0 & -1 & 4 & -1 \\ 0 & \dots & \dots & 0 & -1 & 4 \end{pmatrix} \in R^{(N-1)\times(N-1)} \tag{15}$$

(observe the value of 4 on the main diagonal and the absence of the scaling with h^2) and by $\widehat{T}^h$ the matrix

$$\widehat{T}^h = \begin{pmatrix} h^2 & 0 & 0 \\ 0 & T^h & 0 \\ 0 & 0 & h^2 \end{pmatrix} \in R^{(N+1)\times(N+1)} \,. \tag{16}$$

The matrix

$$A^h = \frac{1}{h^2} \begin{pmatrix} h^2 I_{N+1} & 0 & \dots & \dots & \dots & \dots & 0 \\ 0 & \widehat{T}^h & -\widehat{I}^h & 0 & \dots & \dots & 0 \\ 0 & -\widehat{I}^h & \widehat{T}^h & -\widehat{I}^h & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & 0 & -\widehat{I}^h & \widehat{T}^h & -\widehat{I}^h & 0 \\ 0 & \dots & \dots & 0 & -\widehat{I}^h & \widehat{T}^h & 0 \\ 0 & \dots & \dots & \dots & \dots & 0 & h^2 I_{N+1} \end{pmatrix} \in R^{(N+1)^2\times(N+1)^2} \,. \tag{17}$$

The entries of the vector $\mathbf{f}^h$ need to be overwritten with 0.

Construction of the matrix A^h using the Kronecker Product The matrix A^h can be constructed using the Kronecker product of the identify matrix and the one-dimensional variant of A^h. This construction requires care in properly treating the boundary conditions. We consider a process in four steps.

We first consider the one-dimensional problem $-u''(x) = f(x)$ on the interval $0 < x < 1$ with Dirichlet boundary conditions at both end points. Assume the interval to be subdivided into N intervals such that $N+1$ nodes arise of which $N-1$ are interior nodes. Let $\mathcal{A}^h$ be tridiagonal matrix of size the $(N-1)\times(N-1)$ corresponding to the interior nodes *only*. This matrix can be obtained from the previous assignment by eliminating the first and last row as well as the first and last column.

Next we return to the two-dimensional problem and partition all nodes $1 \leq I \leq (N+1)^2$ into the nodes corresponding to the boundary and the interior points. Let $\mathcal{I}$, $\mathcal{I}_{boundary}$ and $\mathcal{I}_{interior}$ correspond to all the nodes, the nodes on the boundary and the interior points, respectively. Then

$$\mathcal{I} = \mathcal{I}_{boundary} \cup \mathcal{I}_{interior} \text{ and } \mathcal{I}_{boundary} \cap \mathcal{I}_{interior} = \emptyset \,. \tag{18}$$

To construct these sets in Matlab we can use the following piece of code

```
indic  = ones(N+1,N+1); indic(2:end-1,2:end-1) =0;
nnodes = (N+1)^2;
indvec = reshape(indic,[nnodes,1]);
bnd    = find(indvec==1); interior = find(indvec==0);
```

We suggest to try this code on a small example first. In the third step we initialize A^h to the $(N+1)^2 \times N+1)^2$ (corresponding to both interior and boundary nodes) identity matrix as

$$A^h = I_{(N+1)^2} \,. \tag{19}$$

In a fourth and final step we overwrite the equations corresponding to the interior points using

$$A^h(\mathcal{I}_{interior}, \mathcal{I}_{interior}) = \mathcal{A}^h \otimes I_{N-1} + I_{N-1} \otimes \mathcal{A}^h \,, \tag{20}$$

where the first and second term correspond to the discretization of u_{xx} and u_{yy}, respectively. The construction is such that the boundary conditions in A^h are taken into account.

Construction of the vector $\mathbf{f}^h$ using reshape The construction of the vector $\mathbf{f}^h$ requires care in setting the components corresponding the boundary nodes. One possibility is to proceed as follows

1. construct $\mathbf{f}^h$ as a matrix first (instead of directly as a vector) using for instance the `meshgrid` command would do;
2. set the first and last row and column of $\mathbf{f}^h$ as a matrix to zero;
3. reshape $\mathbf{f}^h$ as a matrix into $\mathbf{f}^h$ as a vector using the `reshape` command.

Pen and Paper Assignment 4 Assume $h = 1/2$ and give the size and all elements of the vector $\mathbf{f}^h$.

Pen and Paper Assignment 5 Assume $h = 1/2$ and give the size and all elements of the matrix A^h.

Computer Assignment 6 Build a Matlab code to construct the matrix A^h using the Matlab `kron` command and the matrix $\mathcal{A}^h$. Set $h = 1/2$ and verify your answer of Assignment 4 using your Matlab implementation.

Computer Assignment 7 Build a Matlab code to construct the vector $\mathbf{f}^h$ using the Matlab `reshape` command. Set $h = 1/2$ and verify your answer of Assignment 5 using your Matlab implementation.

Computer Assignment 8 Assume $h = 1/16$, generate A^h and a plot of the non-zero structure of the matrix A^h using the Matlab command `spy`. Explain the non-zero structure of the matrix A^h you observe.

Computer Assignment 9 Assume that $h = 1/16$. Verify that the matrix A^h is symmetric if $\mathcal{A}^h$ is symmetric and if A^h is constructed using the Kronecker product as in (20).

Computer Assignment 10 Assume that $h = 1/4$, $h = 1/8$ and $h = 1/16$. Compute the eigenvalues of A^h using `eig` in Matlab. Verify that the eigenvalues of A^h are **real** and **positive**.

Computer Assignment 11 Assume that $h = 1/(2^p)$, where $p = 1, 2, 3, \ldots$. Compute the solution $\mathbf{u}^h$ using Matlab 's `backslach` $\backslash$ command. Plot the computed solution on the grid. We suggest the combined use of `reshape` and `meshgrid` to make such plots. The use of `reshape` here is intended to convert a vector of length $(N+1)^2$ to a matrix of size $(N+1) \times (N+1)$. Verify that the difference between the analytically given solution and the finite difference computed solution reduces as h becomes smaller.

Vind ik lekker